**IN THE UNITED STATES DISTRICT COURT**
**FOR THE WESTERN DISTRICT OF TEXAS**
**WACO DIVISION**

| | |
|---|---|
| VLSI TECHNOLOGY LLC,<br><br>Plaintiff,<br><br>v.<br><br>INTEL CORPORATION,<br><br>Defendant. | C.A. No. 6:19-cv-00256<br><br>**JURY TRIAL DEMANDED** |

## VLSI TECHNOLOGY LLC'S COMPLAINT FOR PATENT INFRINGEMENT

Plaintiff VLSI Technology LLC ("VLSI"), by and through its undersigned counsel, pleads the following against Intel Corporation ("Intel") and alleges as follows:

## THE PARTIES

1. Plaintiff VLSI is a Delaware limited liability company duly organized and existing under the laws of the State of Delaware. The address of the registered office of VLSI is Corporation Trust Center, 1209 Orange St., Wilmington, DE 19801. The name of VLSI's registered agent at that address is The Corporation Trust Company.

2. VLSI is the assignee and owns all right, title, and interest to U.S. Patent Nos. 7,292,485 ("the '485 Patent"), 7,606,983 ("the '983 Patent"), and 7,793,025 ("the '025 Patent") (collectively, the "Asserted Patents").

3. On information and belief, Defendant Intel is a corporation duly organized and existing under the laws of the State of Delaware, having a regular and established place of business in the Western District of Texas, including at 1300 S. Mopac Expressway, Austin, Texas 78746.[1]

## JURISDICTION AND VENUE

4. This is an action arising under the patent laws of the United States, 35 U.S.C. § 1 *et seq*. Accordingly, this Court has subject matter jurisdiction pursuant to 28 U.S.C. §§ 1331 and 1338(a).

5. This Court has personal jurisdiction over Intel because Intel manufactures products that are and have been used, offered for sale, sold, and purchased in the Western District of Texas, and Intel has committed, and continues to commit, acts of infringement in the Western District of Texas, has conducted business in the Western District of Texas, and/or has engaged in continuous and systematic activities in the Western District of Texas.

6. Under 28 U.S.C. §§ 1391(b)-(d) and 1400(b), venue is proper in this judicial district because Intel maintains a regular and established place of business in this district and has committed acts of infringement within this judicial district giving rise to this action.

[1] https://www.intel.com/content/www/us/en/location/usa.html; https://www.intel.com/content/www/us/en/corporate-responsibility/intel-in-texas.html.

**FIRST CLAIM**

**(Infringement of U.S. Patent No. 7,292,485)**

7. VLSI re-alleges and incorporates herein by reference Paragraphs 1-6 of its Complaint.

8. The '485 Patent, entitled "SRAM having variable power supply and method therefor," was duly and lawfully issued November 6, 2007. A true and correct copy of the '485 Patent is attached hereto as Exhibit 1.

9. The '485 Patent names Olga R. Lu, Lawrence F. Childs, and Craig D. Gunderson as co-inventors.

10. The '485 Patent has been in full force and effect since its issuance. VLSI owns by assignment the entire right, title, and interest in and to the '485 Patent, including the right to seek damages for past, current, and future infringement thereof.

11. The '485 Patent states that it "relates generally to memories, and more particularly, to a static random access (SRAM) memory having a variable power supply and method therefor." Ex. 1 at 1:6-9.

12. The '485 Patent explains that "[s]tatic random access memories (SRAMs) are generally used in applications requiring high speed, such as memory in a data processing system." *Id*. at 1:13-15. The '485 Patent explains "increasing [certain] ratios improves cell stability. However, improving stability comes at the expense of lower write performance." *Id*. at 1:36-38. The patent further explains, "there is a need for a SRAM having improved cell stability while also having improved write margins." *Id*. at 1:42-43.

13. VLSI is informed and believes, and thereon alleges, that Intel has infringed and unless enjoined will continue to infringe one or more claims of the '485 Patent, in violation of 35

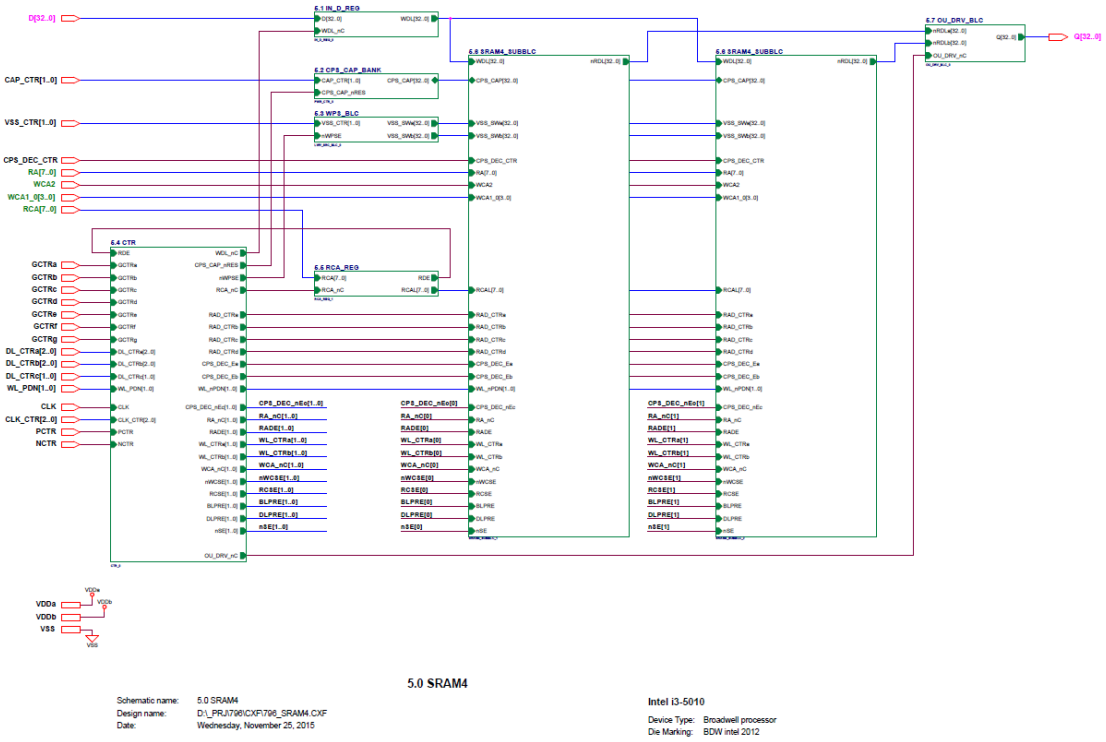
U.S.C. § 271, by, among other things, making, using, offering to sell, and selling within the United States, supplying or causing to be supplied in or from the United States, and importing into the United States, without authority or license, Intel products that use infringing write-assist technology in static random access memory ("SRAM") arrays.

14. For example, the '485 accused products embody every limitation of at least claim 12 of the '485 Patent, literally or under the doctrine of equivalents, as set forth below. The further descriptions below, which are based on publicly available information, are preliminary examples and are non-limiting.

**["12. A method, comprising: providing a memory comprising:"]**

15. Intel Broadwell processors operate using a method comprising providing a memory.

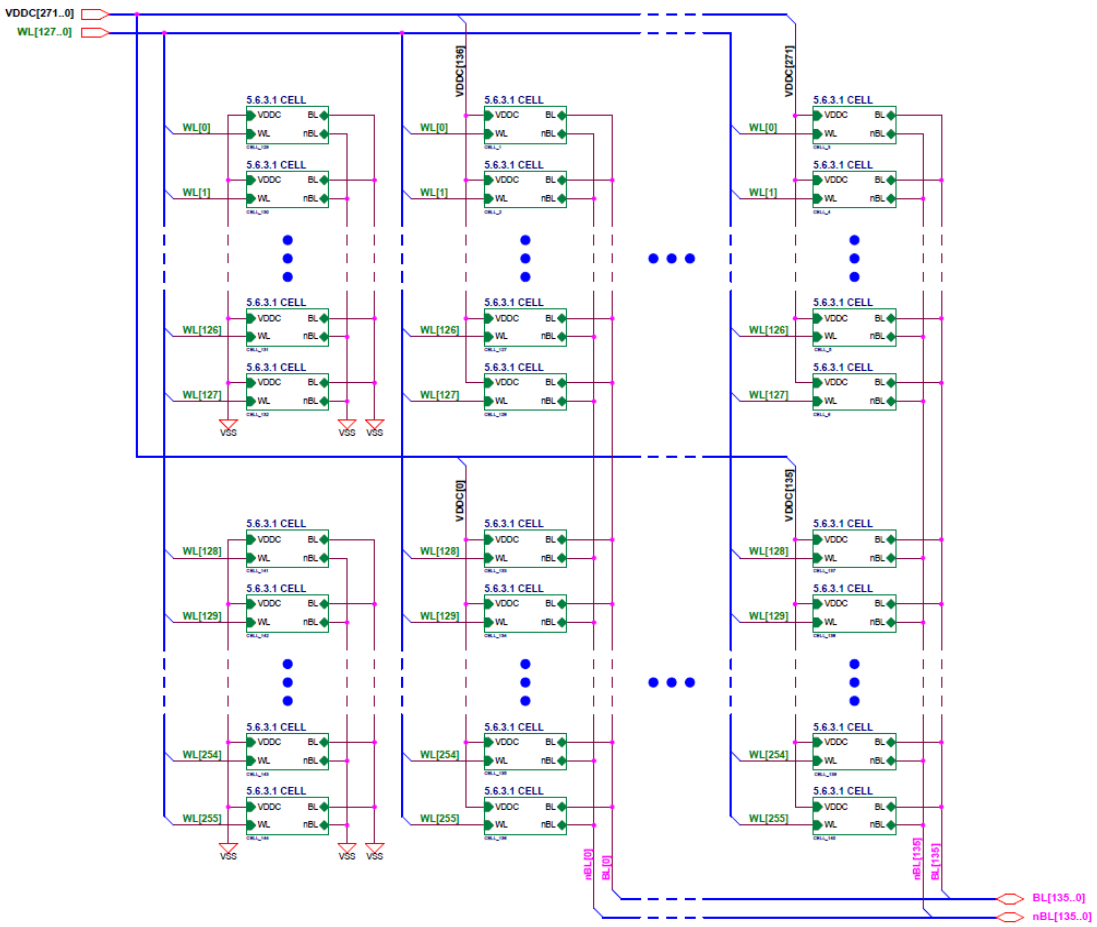
16. For example, reverse engineering shows that Broadwell processors include several SRAM arrays, including the array shown below as SRAM4:




**[“a memory array comprising a first line of memory cells and a second line of memory cells;”]**

17. Intel Broadwell processors operate using a method comprising providing a memory array comprising a first line of memory cells and a second line of memory cells.

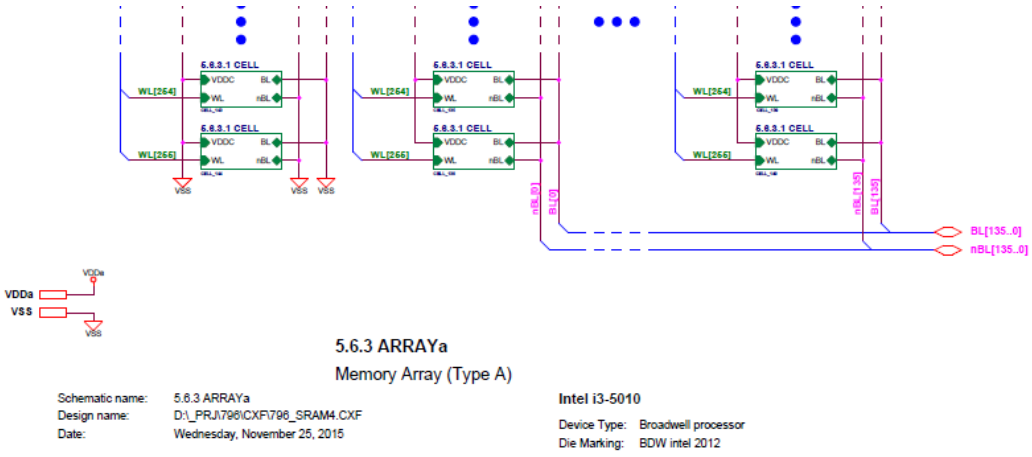
18. For example, reverse engineering shows that Broadwell's SRAM arrays are constructed using a plurality of columns:




**[“a first power supply terminal;”]**

19. Intel Broadwell processors operate using a method comprising providing a first power supply terminal.

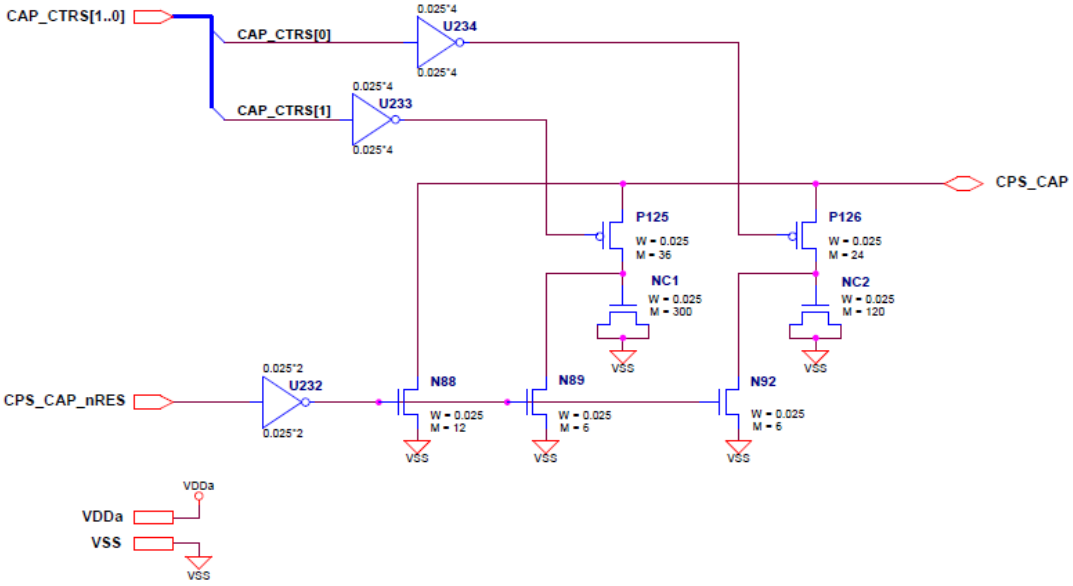
20. For example, reverse engineering shows that Broadwell includes a first power supply terminal (e.g., VDDa):




**[“a first capacitance structure includes a plurality of dummy cells;”]**

21. Intel Broadwell processors operate using a method comprising providing a first capacitance structure that includes a plurality of dummy cells.

22. For example, reverse engineering shows that Intel Broadwell processors include a cell power switch capacitor comprising a plurality of dummy transistors (*e.g.*, NC1 and NC2):



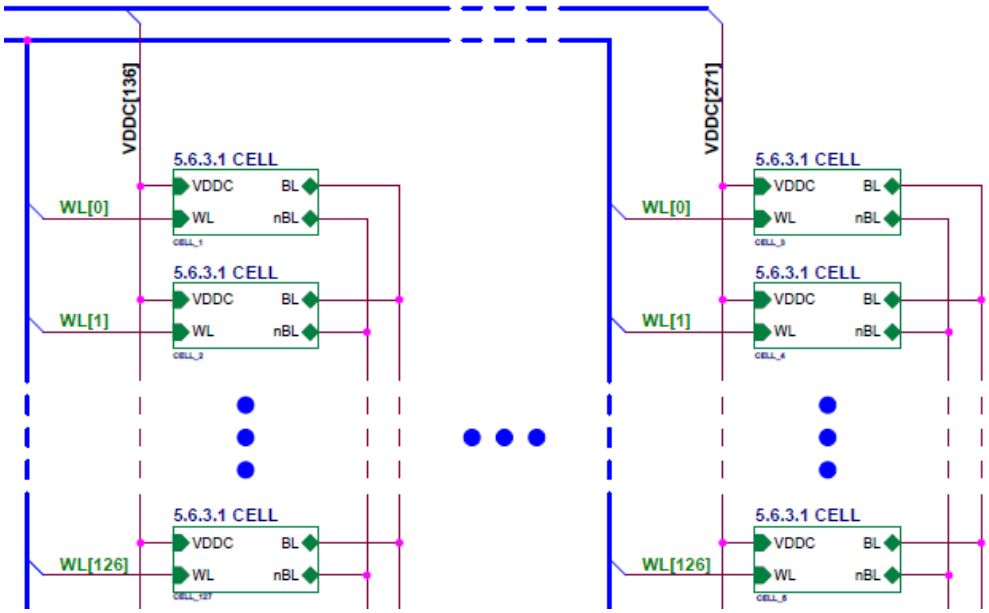

**5.2.2 CPS_CAP**

Cell Power Switch Capacitor

**[“a first power supply line coupled to the first line of memory cells; and”]**

23. Intel Broadwell processors operate using a method comprising providing a first power supply line coupled to the first line of memory cells.

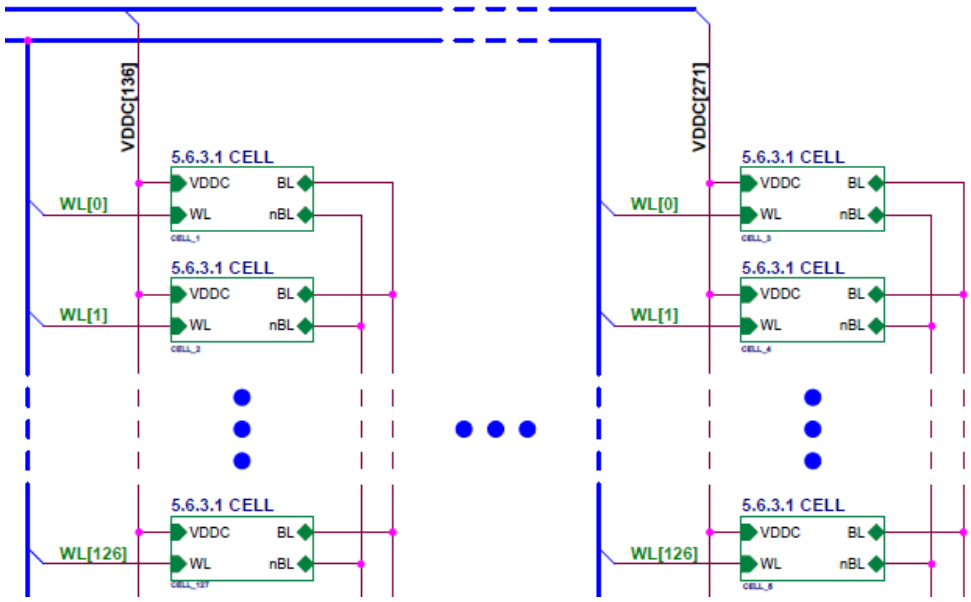
24. For example, reverse engineering shows that Broadwell includes a first power supply line (*e.g.*, VDDC[136]), coupled to the first line of memory cells:




**[“a second power supply line coupled to the second line of memory cells;”]**

25. Intel Broadwell processors operate using a method comprising providing a second power supply line coupled to the second line of memory cells.

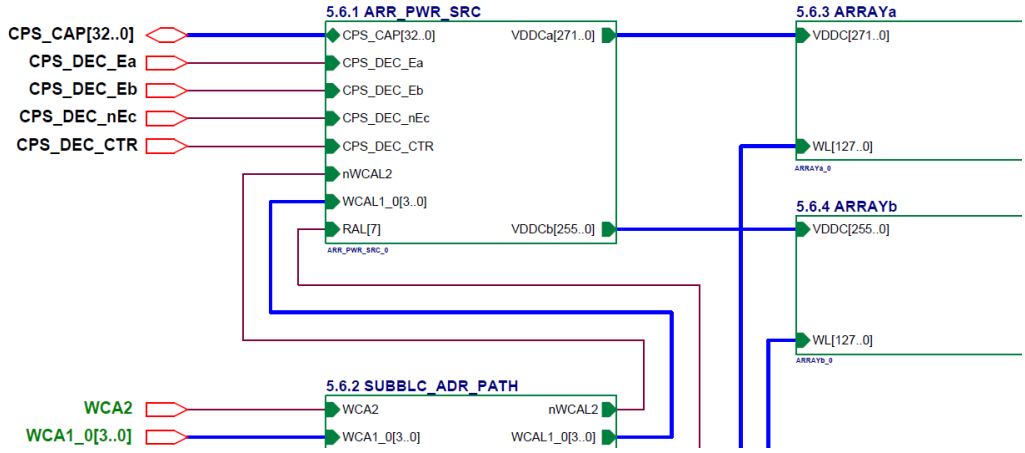
26. For example, reverse engineering shows that Broadwell includes a second power supply line (*e.g.*, VDDC[271]) coupled to the second line of memory cells:



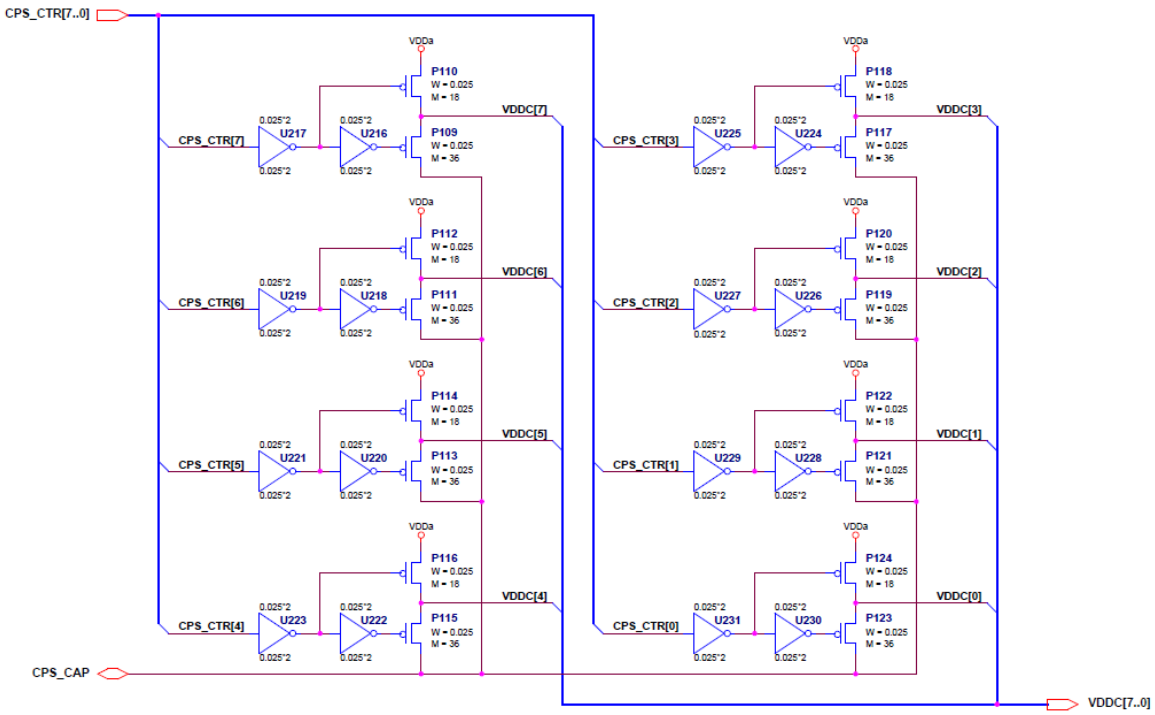

**[“a switching circuit that has transistors that connected between the first power supply terminal, the first power supply line, the second power supply line and the first capacitance structure”]**

27. Intel Broadwell processors operate using a method comprising providing a switching circuit that has transistors that connected between the first power supply terminal, the first power supply line, the second power supply line and the first capacitance structure.

28. For example, reverse engineering shows that Broadwell includes a switching circuit, *e.g.*, ARR_PWR_SRC, that has transistors that connect a capacitance structure, the power supply lines, and the plurality of power supply lines:




29. Reverse engineering also shows connection of first power supply terminal (VDDa) to VDDc:



**["selecting the second line of memory cells for writing;"]**

30. Intel Broadwell processors operate using a method comprising selecting the second line of memory cells for writing.

31. For example, the column based voltage bias technique used in the SRAM selects particular columns for reading. *See, e.g.*, "A 0.6V 1.5GHz 84Mb SRAM Design in 14nm FinFET CMOS Technology" [hereinafter "SRAM"] at 1: "The CS-TVC operation begins with a self-timed pulse (TVC PULSE) aligned to the rising edge of WL that simultaneously disconnects the selected VCS region from VCC, disables the NMOS devices in the CS-TVC capacitor, and connects VCS to the pre-discharged GCSCAP node."

**["coupling the first power supply terminal to the first power supply line;"]**

32. Intel Broadwell processors operate using a method comprising coupling the first power supply terminal to the first power supply line.

33. For example, columns that are not being written to (including the first column) are connected to the first power supply terminal to their power supply lines. This power supply terminal is described as "VCS" by Intel. *See, e.g.*, SRAM at 1: "The CS-TVC operation begins with a self-timed pulse (TVC PULSE) aligned to the rising edge of WL that simultaneously disconnects the selected VCS region from VCC, disables the NMOS devices in the CS-TVC capacitor, and connects VCS to the pre-discharged GCSCAP node. Charge is balanced through a PMOS switch between the VCS region selected and GCSCAP, resulting in a temporary suppression of the VCS node to improve write margin. The falling edge of the TVC pulse completes the operation and restores the VCS voltage level to VCC."

**["decoupling the second line of memory cells from the first power supply terminal;"]**

34. Intel Broadwell processors operate using a method comprising decoupling the second line of memory cells from the first power supply terminal.

35. For example, columns that are being written to (including the second column) are disconnected from the first power supply terminal. *See, e.g.*, SRAM at 1: "The CS-TVC operation begins with a self-timed pulse (TVC PULSE) aligned to the rising edge of WL that simultaneously disconnects the selected VCS region from VCC, disables the NMOS devices in the CS-TVC capacitor, and connects VCS to the pre-discharged GCSCAP node. Charge is balanced through a PMOS switch between the VCS region selected and GCSCAP, resulting in a temporary suppression of the VCS node to improve write margin. The falling edge of the TVC pulse completes the operation and restores the VCS voltage level to VCC."

**["coupling charge from the second power supply line to the first capacitance structure; and"]**

36. Intel Broadwell processors operate using a method comprising coupling charge from the second power supply line to the first capacitance structure.

37. For example, columns that are being written to (including the second column) have charge coupled to the capacitance structure, here referred to as GCSCAP. *See, e.g.*, SRAM at 1: "The CS-TVC operation begins with a self-timed pulse (TVC PULSE) aligned to the rising edge of WL that simultaneously disconnects the selected VCS region from VCC, disables the NMOS devices in the CS-TVC capacitor, and connects VCS to the pre-discharged GCSCAP node. Charge is balanced through a PMOS switch between the VCS region selected and GCSCAP, resulting in

a temporary suppression of the VCS node to improve write margin. The falling edge of the TVC pulse completes the operation and restores the VCS voltage level to VCC."

**["writing a memory cell in the second line of memory cells."]**

38. Intel Broadwell processors operate using a method comprising writing a memory cell in the second line of memory cells.

39. For example, memory cells are written in columns that are being written to (including the second column). *See, e.g.*, SRAM at 1: "The CS-TVC operation begins with a self-timed pulse (TVC PULSE) aligned to the rising edge of WL that simultaneously disconnects the selected VCS region from VCC, disables the NMOS devices in the CS-TVC capacitor, and connects VCS to the pre-discharged GCSCAP node. Charge is balanced through a PMOS switch between the VCS region selected and GCSCAP, resulting in a temporary suppression of the VCS node to improve write margin. The falling edge of the TVC pulse completes the operation and restores the VCS voltage level to VCC."

40. Intel has had knowledge of the '485 Patent and its infringement of the '485 Patent at least since the filing of the complaint in *VLSI Technology LLC v. Intel Corp.*, Civil Action No. 19-00426 (D. Del.) (filed Mar. 1, 2019) (the "Delaware Complaint") which asserted infringement by Intel of the '485 Patent, and if it did not have actual knowledge prior to that time, it was willfully blind to the existence of the '485 Patent and its infringement of the '485 Patent based on, for example, its publicly-known corporate policy forbidding its employees from reading patents held by outside companies or individuals. For example, in *Intel Corp. v. Future Link Sys., LLC*, 268 F. Supp. 3d 605, 623 (D. Del. 2017) the court noted the patent owner's observation that "Intel's own engineers concede that they avoid reviewing other, non-Intel patents so as to avoid willfully infringing them." As a further example, former Intel employees, including Intel's long-time Chief

Architect Robert Colwell, have admitted that this policy's purpose is to "avoid possible triple damages for 'willful infringement.'" As still another example, on information and belief, Intel has been sued for infringing patents previously assigned to NXP while this policy was in place. Under the circumstances present here, including explicit notice having been provided of Intel's infringement of other NXP patents and NXP's competitive position with Intel in the marketplace, Intel knew or should have known of the high probability that NXP had patented other technologies, such as those to which the '485 Patent is directed, that Intel had included within its microprocessor products. As yet another example, on information and belief, Olga Lu, an inventor on the '485 Patent, has been working at Intel since at least January 2011 and currently works at Intel as an SRAM/FUSE design manager, and as such Intel has been aware of the '485 Patent and its infringement of the '485 Patent since at least the date Ms. Lu began working at Intel. Intel should have known that its conduct was infringing both prior to and following the filing of VLSI's Delaware Complaint.

41. VLSI is informed and believes, and thereon alleges, that Intel actively, knowingly, and intentionally has induced infringement of the '485 Patent by, for example, controlling the design and manufacture of, offering for sale, selling, supplying, and otherwise providing instruction and guidance regarding the above-described products with the knowledge and specific intent to encourage and facilitate infringing uses of such products by its customers both inside and outside the United States. For example, Intel publicly provides documentation, including datasheets available through Intel's publicly accessible ARK service and software developer's manuals, instructing customers on uses of Intel's products that infringe the methods of the '485 Patent. *See, e.g.*, http://ark.intel.com. On information and belief, Intel's customers directly infringe the '485 Patent by, for example, making, using, offering to sell, and selling within the

United States, and importing into the United States, without authority or license, products containing the above-described Intel products.

42. VLSI is informed and believes, and thereon alleges, that Intel has contributed to the infringement by its customers of the ’485 Patent by, without authority, importing, selling and offering to sell within the United States materials and apparatuses for practicing the claimed invention of the ’485 Patent both inside and outside the United States. For example, the above-described products constitute a material part of the inventions of the ’485 Patent and are not staple articles or commodities of commerce suitable for substantial noninfringing use. On information and belief, Intel knows that the above-described products constitute a material part of the inventions of the ’485 Patent and are not staple articles or commodities of commerce suitable for substantial noninfringing use. On information and belief, Intel’s customers directly infringe the ’485 Patent by, for example, making, using, offering to sell, and selling within the United States, and importing into the United States, without authority or license, products containing the above-described Intel products.

43. As a result of Intel’s infringement of the ’485 Patent, VLSI has been damaged. VLSI is entitled to recover for damages sustained as a result of Intel’s wrongful acts in an amount subject to proof at trial.

44. To the extent 35 U.S.C. § 287 is determined to be applicable, on information and belief its requirements have been satisfied with respect to the ’485 Patent.

45. In addition, Intel’s infringing acts and practices have caused and are causing immediate and irreparable harm to VLSI.

46. VLSI is informed and believes, and thereon alleges, that Intel’s infringement of the ’485 Patent has been and continues to be willful. As noted above, Intel has had knowledge of the

'485 Patent and its infringement of the '485 Patent. Intel has deliberately continued to infringe in a wanton, malicious, and egregious manner, with reckless disregard for VLSI's patent rights. Thus, Intel's infringing actions have been and continue to be consciously wrongful.

47. Based on the information alleged in this claim, as well as the information alleged in the Second and Third Claims *infra*, VLSI is informed and believes, and thereon alleges, that this is an exceptional case, which warrants an award of attorney's fees to VLSI pursuant to 35 U.S.C. § 285.

## SECOND CLAIM

**(Infringement of U.S. Patent No. 7,606,983)**

48. VLSI re-alleges and incorporates herein by reference Paragraphs 1-47 of its Complaint.

49. The '983 Patent, entitled "Sequential ordering of transactions in digital systems with multiple requestors," was duly and lawfully issued October 20, 2009. A true and correct copy of the '983 Patent is attached hereto as Exhibit 2.

50. The '983 Patent names Kevin Locker as the inventor.

51. The '983 Patent has been in full force and effect since its issuance. VLSI owns by assignment the entire right, title, and interest in and to the '983 Patent, including the right to seek damages for past, current, and future infringement thereof.

52. The '983 Patent "relates generally to methods and apparatuses for designing digital systems." Ex. 2 at 1:7-8.

53. The '983 Patent explains that "[m]any digital electronic devices include multiple autonomous or semi-autonomous functional modules, such as processors, that share access to common resources, such as memory." *Id*. at 1:19-22. The '983 Patent describes, for example,

providing "an improved transaction ordering policy in which individual occurrences of access requests can specify whether or not the associated transaction is to be performed in order." *Id*. at 4:57-60.

54. VLSI is informed and believes, and thereon alleges, that Intel has infringed and unless enjoined will continue to infringe one or more claims of the '983 Patent, in violation of 35 U.S.C. § 271, by, among other things, making, using, offering to sell, and selling within the United States, supplying or causing to be supplied in or from the United States, and importing into the United States, without authority or license, Intel products that implement the Intel Quick Path Interconnect ("QPI") Link Layer in an infringing manner.

55. For example, the '983 accused products embody every limitation of at least claim 11 of the '983 Patent, literally or under the doctrine of equivalents, as set forth below. The further descriptions below, which are based on publicly available information, are preliminary examples and are non-limiting.

**["11. A method of processing information, the method comprising:"]**

56. Broadwell Server processors, which support the Intel QPI Link Layer, use a method of processing information as shown below. *See, e.g.*, https://ark.intel.com/products/91317/Intel-Xeon-Processor-E5-2699-v4-55M-Cache-2-20-GHz-:

**Essentials** Export specifications

| | |
|---|---|
| Product Collection | Intel® Xeon® Processor E5 v4 Family |
| Code Name | Products formerly Broadwell |
| Vertical Segment | Server |
| Processor Number | E5-2699V4 |
| Status | Launched |
| Launch Date | Q1'16 |
| Lithography | 14 nm |
| Recommended Customer Price | $4115.00 |

**Performance**

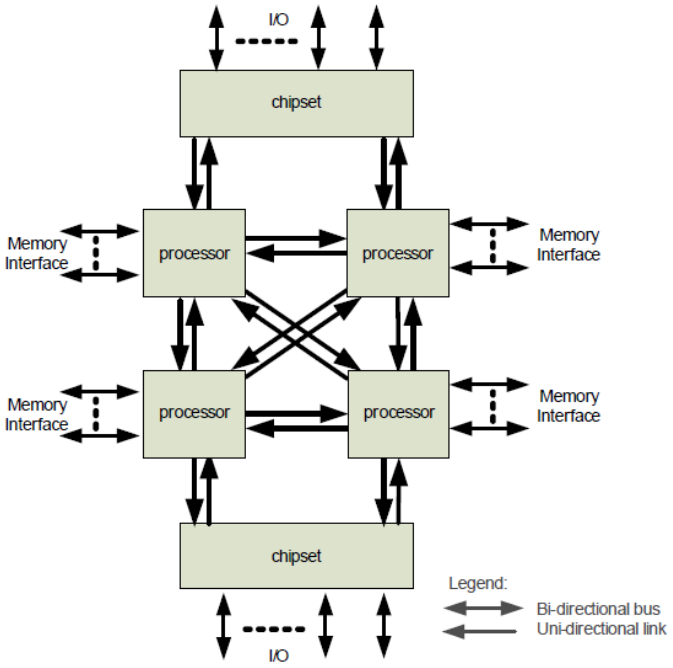
| | |
|---|---|
| # of Cores | 22 |
| # of Threads | 44 |
| Processor Base Frequency | 2.20 GHz |
| Max Turbo Frequency | 3.60 GHz |
| Cache | 55 MB SmartCache |
| Bus Speed | 9.6 GT/s QPI |
| # of QPI Links | 2 |
| TDP | 145 W |
| VID Voltage Range | 0 |

**["holding data;"]**

57. Broadwell Server processors use a method of processing information comprising holding data.

58. For example, Broadwell Server processors are each connected via a memory interface to a memory controller. *See, e.g.*, An Introduction to the Intel QuickPath Interconnect [hereinafter "QPI"] at 8:

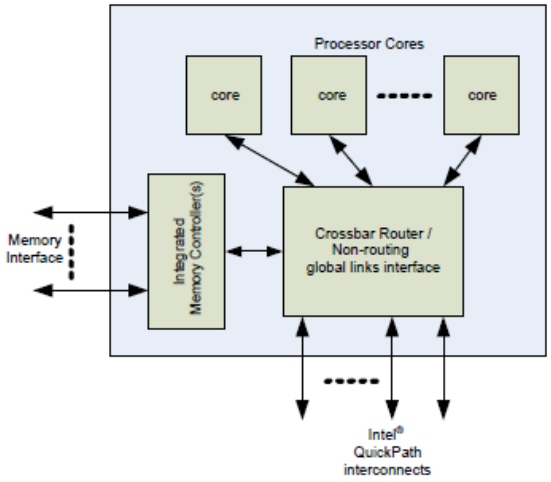
**Figure 6. Intel® QuickPath Interconnect**



59. As another example, Broadwell Server processors include shared and separate caches. *See, e.g.*, QPI at 8:

Figure 7 shows a schematic of a processor with external Intel® QuickPath Interconnects. The processor may have one or more cores. When multiple cores are present, they may share caches or have separate caches. The processor also typically has one or more integrated memory controllers. Based on the level of scalability supported in the processor, it may include an integrated crossbar router and more than one Intel® QuickPath Interconnect port (a port contains a pair of uni-directional links).

**Figure 7. Block Diagram of Processor with Intel® QuickPath Interconnects**



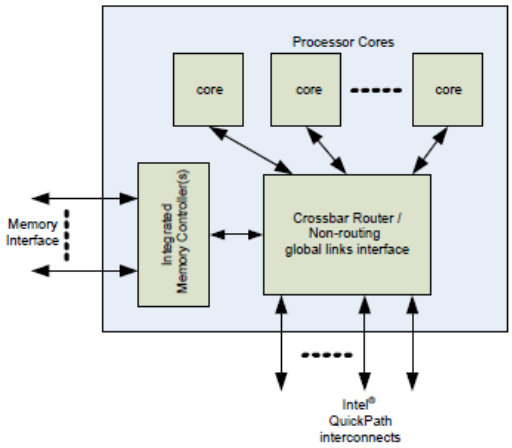

**[“performing in parallel at least two series of operations;”]**

60. Broadwell Server processors use a method of processing information comprising performing in parallel at least two series of operations.

61. For example, Broadwell Server systems contain multiple cores that perform operations in parallel. *See, e.g.*, QPI at 8:

Figure 7 shows a schematic of a processor with external Intel® QuickPath Interconnects. The processor may have one or more cores. When multiple cores are present, they may share caches or have separate caches. The processor also typically has one or more integrated memory controllers. Based on the level of scalability supported in the processor, it may include an integrated crossbar router and more than one Intel® QuickPath Interconnect port (a port contains a pair of uni-directional links).

**Figure 7. Block Diagram of Processor with Intel® QuickPath Interconnects**




**[“generating an access request when performing one of the operations when the operation involves accessing the data;”]**

62. Broadwell Server processors use a method of processing information comprising generating an access request when performing one of the operations when the operation involves accessing the data.

63. For example, when one core in a Broadwell Server system makes an access to data, such as a read from memory, it may create one or more access requests such as, for example, a

message in a Home Message Class or a message in a Non-coherent Standard Message Class. *See*, *e.g.*, "Weaving High Performance Multiprocessor Fabric" [hereinafter "Fabric"] at 42, 119-120:

**Request Generation**

Whenever an Intel QPI agent is required to read from, or write data to memory or I/O, the agent creates an internal request to the appropriate address space. A request bound for coherent shared memory first checks the local cache for a valid copy of the data. If that check fails to find the data in the agent's local cache, the protocol layer uses the rules of the Intel QPI cache coherence protocol to obtain the most up-to-date copy of data from the system memory, or from another caching agent in the system. The protocol layer then issues a set of requests and associated responses to complete the operations.

Whenever an agent generates a request, either from the protocol layer or directly from the processor core for non-coherent operations, the request needs to undergo a mapping operation in order to determine its destination on the Intel QPI fabric. Agents generating a request specify the destination in terms of the system address space, but destinations on Intel QPI are determined by a Node ID field embedded in the packets being sent. A piece of logic that exists in every requesting agent is known as the *source address decoder* (SAD). The SAD logic takes into account the nature of messages that are to be issued, uses a lookup table to map from system address to Node ID, and then provides that information to be included in the messages sent out onto the Intel QPI fabric.

*Home Message Class—HOM*

HOM class traffic consists of requests for access to the coherent data space, or for snoop responses related to those requests and to the SNP messages. HOM traffic is always directed to the home agent responsible for that particular address in the coherent memory space. Requests are always generated by a caching agent. Snoop responses are also always generated by the caching agent. HOM messages are also the minimum packet size of 1 or 2 flits.

In current products, the ordering requirement placed upon HOM messages is as follows. For any given caching agent (source) and home agent (destination) combination, messages are kept in order on a per-address basis. Messages from different sources or being sent to different destinations have no ordering requirements respective to each other. Likewise messages to different addresses have no ordering requirements relative to each other. Current home agent designs rely upon the link layer maintaining this HOM channel ordering to ensure proper operation of the coherent protocol.

*Non-Coherent Standard—NCS*

NCS messages are requests dealing with access to system resources that are generally outside of the coherent memory space. This includes things such as reads to non-coherent memory regions, read and write configuration and I/O space transactions, and interrupt acknowledgements. Certain NCS transactions can also access the coherent memory space, but it is not required that coherency be maintained by the Intel QPI hardware in this case. Such read requests would be performed without the addition of snoop requests and responses. Small packets are used, ranging from the minimum packet size, up to three flit packet sizes for some message types. NCS packets do not contain any data flits. However, some messages in NCS support up to an 8 byte payload embedded into the packet's header flits.

**["including in each access request an indication of whether or not this occurrence of the access request has a specified order among other occurrences of the access request; including an indication of the specified order in those occurrence[s] of the access request that ar[e] ordered; receiving the access requests from each of the series of operations;"]**

64. Broadwell Server processors use a method of processing information comprising including in each access request an indication of whether or not this occurrence of the access request has a specified order among other occurrences of the access request, including an indication of the specified order in those occurrences of the access request that are ordered; receiving the access requests from each of the series of operations.

65. For example, each request sent over the QuickPath Interconnect in connection with Broadwell Server processors contains indications including, *e.g.*, Requester Transaction ID, address, and type that indicates that certain types of requests must be ordered, and if they are to be

ordered, which order they are to be performed in. These ordering requirements are based, *e.g.*, on "stall" rules that require that certain messages be sent before others. *See, e.g.*, Datasheet at 81, 84:

## Outgoing Request Buffer (ORB)

When a transaction is issued onto the Intel QuickPath Interconnect, an ORB entry is allocated. This list keeps all pertinent information about the transaction header needed to complete the request. It also stores the cacheline address for coherent transactions to allow conflict checking with snoops and other local requests. See Section 4.10 for details. An Intel QuickPath Interconnect response may come as multiple phases. The ORB tracks each completion phase (that is, Data* and Cmp) and any conflicts (that is, snoops, FrcAckCnflt Response) in the ORB.

When a request is issued, a RTID (Requestor Transaction ID) is assigned based on Home NodeID. Limitations are placed on the RTID based on how many transactions each home agent can support. The RTID allocation limitation binds the tag to a specific range; this range is referred to as *MaxRequests*, where the range is 0 to *MaxRequests-1*. This is value is programmable in the IOH configuration registers.

The Home NodeID and RTID are returned in the responses. The ORB must provide a way to associate the response's RTID and Home NodeID with a ORB entry. This can be done through a reverse lookup table based on RTID and Home NodeID or by searching the table for matching RTID and NodeID. Simplification of the reverse lookup function is possible by limiting the scope of the RTID allocation as described in Section 4.9.2.

### ORB Depth

The ORB depth is based on the number of requests the IOH needs to be pending on Intel QuickPath Interconnect to achieve full bandwidth, given worst-case average latency for a 4S system. This latency is calculated from allocation and de-allocation of tag. The ORB depth is 256 entries. The IOH will further partition the 256 entries into 128 entries per port. For evenly distributed traffic, this will provide the equivalent of 256 entries.

**Table 4-14. Local-Local Conflict Actions**

| Local Request | ORB or Write Cache State | Action |
|---|---|---|
| Rd or NonSnpRd or NonSnpWr* | Rd or NonSnpRd or NonSnpWr* | • Stall until completed. |
| | RFO | • Stall until EWB completed.<br>• Force EWB on Promotion. |
| | EWB | • Stall until EWB completed. |
| | E- or MG-state | • Stall until EWB completed.<br>• Force EWB on Promotion. |
| | M-state | • Stall until EWB completed.<br>• Force EWB. |
| | Non-Coh VT-d table data | • Data may be snarfed locally for other Intel VT-d Reads. |
| RFO | Rd or NonSnpRd or NonSnpWr* | • Stall until completed. |
| | RFO | • Stall until Promotion.<br>• If promotion does not evict the line then Complete RFO.<br>• If promotion causes EWB, then send RFO after EWB completed. |
| | EWB | • Stall until EWB completed. |
| | E- or MG-state | • Stall until Promotion.<br>• If promotion does not evict the line then Complete RFO.<br>• If promotion causes EWB, then send RFO after EWB completed. |
| | M-state | • Complete immediately, no stall condition. |
| M-Promote | Rd or NonSnpRd or NonSnpWr* or RFO or EQB or M-state | • Impossible. EWB can only be received in E or MG state. |
| | E- or MG-state | • Normal flow to M-state and Eviction.<br>• If M-state does not cause eviction under normal rules then Conflict queue is checked to see if EWB required or RFO completion required.<br>• EWB: on EWB completion, next conflict is cleared.<br>• If multiple ownerships are granted simultaneously, then state will change to MG state. |

66. *See also, e.g.*, Fabric at 118-120:

**Table 3.1** Message Classes

| Message Class | Abbreviation | Ordering Requirements | Data Content? |
|---|---|---|---|
| Snoop | SNP | None | No |
| Home | HOM | Limited | No |
| Data Response | DRS | None | Yes |
| Non-Data Response | NDR | None | No |
| Non-coherent Standard | NCS | None | No1 |
| Non-coherent Bypass | NCB | None | Yes |
| Special (Link Management) | SPC | None | No |

1 See section on NCS for more information

*Home Message Class—HOM*

HOM class traffic consists of requests for access to the coherent data space, or for snoop responses related to those requests and to the SNP messages. HOM traffic is always directed to the home agent responsible for that particular address in the coherent memory space. Requests are always generated by a caching agent. Snoop responses are also always generated by the caching agent. HOM messages are also the minimum packet size of 1 or 2 flits.

In current products, the ordering requirement placed upon HOM messages is as follows. For any given caching agent (source) and home agent (destination) combination, messages are kept in order on a per-address basis. Messages from different sources or being sent to different destinations have no ordering requirements respective to each other. Likewise messages to different addresses have no ordering requirements relative to each other. Current home agent designs rely upon the link layer maintaining this HOM channel ordering to ensure proper operation of the coherent protocol.

*Non-Coherent Standard—NCS*

NCS messages are requests dealing with access to system resources that are generally outside of the coherent memory space. This includes things such as reads to non-coherent memory regions, read and write configuration and I/O space transactions, and interrupt acknowledgements. Certain NCS transactions can also access the coherent memory space, but it is not required that coherency be maintained by the Intel QPI hardware in this case. Such read requests would be performed without the addition of snoop requests and responses. Small packets are used, ranging from the minimum packet size, up to three flit packet sizes for some message types. NCS packets do not contain any data flits. However, some messages in NCS support up to an 8 byte payload embedded into the packet's header flits.

**["determining a performance order for the access requests, wherein if the access requests are ordered then the performance order conforms to the specified order; and performing the access requests in the performance order."]**

67. Broadwell Server processors use a method of processing information comprising determining a performance order for the access requests, wherein if the access requests are ordered then the performance order conforms to the specified order and performing the access requests in the performance order.

68. For example, QPI agents in Broadwell Server processors perform the access requests in the order as determined by the rules discussed above. *See, e.g.*, Datasheet at 81, 84:

## Outgoing Request Buffer (ORB)

When a transaction is issued onto the Intel QuickPath Interconnect, an ORB entry is allocated. This list keeps all pertinent information about the transaction header needed to complete the request. It also stores the cacheline address for coherent transactions to allow conflict checking with snoops and other local requests. See Section 4.10 for details. An Intel QuickPath Interconnect response may come as multiple phases. The ORB tracks each completion phase (that is, Data* and Cmp) and any conflicts (that is, snoops, FrcAckCnflt Response) in the ORB.

When a request is issued, a RTID (Requestor Transaction ID) is assigned based on Home NodeID. Limitations are placed on the RTID based on how many transactions each home agent can support. The RTID allocation limitation binds the tag to a specific range; this range is referred to as *MaxRequests*, where the range is 0 to *MaxRequests-1*. This is value is programmable in the IOH configuration registers.

The Home NodeID and RTID are returned in the responses. The ORB must provide a way to associate the response's RTID and Home NodeID with a ORB entry. This can be done through a reverse lookup table based on RTID and Home NodeID or by searching the table for matching RTID and NodeID. Simplification of the reverse lookup function is possible by limiting the scope of the RTID allocation as described in Section 4.9.2.

### ORB Depth

The ORB depth is based on the number of requests the IOH needs to be pending on Intel QuickPath Interconnect to achieve full bandwidth, given worst-case average latency for a 4S system. This latency is calculated from allocation and de-allocation of tag. The ORB depth is 256 entries. The IOH will further partition the 256 entries into 128 entries per port. For evenly distributed traffic, this will provide the equivalent of 256 entries.

**Table 4-14. Local-Local Conflict Actions**

| Local Request | ORB or Write Cache State | Action |
|---|---|---|
| Rd or NonSnpRd or NonSnpWr* | Rd or NonSnpRd or NonSnpWr* | • Stall until completed. |
| | RFO | • Stall until EWB completed.<br>• Force EWB on Promotion. |
| | EWB | • Stall until EWB completed. |
| | E- or MG-state | • Stall until EWB completed.<br>• Force EWB on Promotion. |
| | M-state | • Stall until EWB completed.<br>• Force EWB. |
| | Non-Coh VT-d table data | • Data may be snarfed locally for other Intel VT-d Reads. |
| RFO | Rd or NonSnpRd or NonSnpWr* | • Stall until completed. |
| | RFO | • Stall until Promotion.<br>• If promotion does not evict the line then Complete RFO.<br>• If promotion causes EWB, then send RFO after EWB completed. |
| | EWB | • Stall until EWB completed. |
| | E- or MG-state | • Stall until Promotion.<br>• If promotion does not evict the line then Complete RFO.<br>• If promotion causes EWB, then send RFO after EWB completed. |
| | M-state | • Complete immediately, no stall condition. |
| M-Promote | Rd or NonSnpRd or NonSnpWr* or RFO or EQB or M-state | • Impossible. EWB can only be received in E or MG state. |
| | E- or MG-state | • Normal flow to M-state and Eviction.<br>• If M-state does not cause eviction under normal rules then Conflict queue is checked to see if EWB required or RFO completion required.<br>• EWB: on EWB completion, next conflict is cleared.<br>• If multiple ownerships are granted simultaneously, then state will change to MG state. |

69. *See also, e.g.*, QPI at 109:

**Table 6-1. Ordering Term Definitions**

| Term | Definition |
|---|---|
| Intel QuickPath Interconnect Ordering Domain | The Intel QuickPath Interconnect has a relaxed ordering model allowing reads, writes and completions to flow independent of each other. Intel QuickPath Interconnect implements this through the use of multiple, independent virtual channels. In general, the Intel QuickPath Interconnect ordering domain is considered unordered. |
| PCI Express Ordering Domain | PCI Express (and all other prior PCI generations) have specific ordering rules to enable low cost components to support the Producer-Consumer model. For example, no transaction can pass a write flowing in the same direction. In addition, PCI implements ordering relaxations to avoid deadlocks (for example, completions must pass non-posted requests). The set of these rules are described in *PCI Express Base Specification*, Revision 1.0a. |

70. *See also, e.g.*, Fabric at 118-120:

**Table 3.1** Message Classes

| Message Class | Abbreviation | Ordering Requirements | Data Content? |
|---|---|---|---|
| Snoop | SNP | None | No |
| Home | HOM | Limited | No |
| Data Response | DRS | None | Yes |
| Non-Data Response | NDR | None | No |
| Non-coherent Standard | NCS | None | No1 |
| Non-coherent Bypass | NCB | None | Yes |
| Special (Link Management) | SPC | None | No |

1 See section on NCS for more information

*Home Message Class—HOM*

HOM class traffic consists of requests for access to the coherent data space, or for snoop responses related to those requests and to the SNP messages. HOM traffic is always directed to the home agent responsible for that particular address in the coherent memory space. Requests are always generated by a caching agent. Snoop responses are also always generated by the caching agent. HOM messages are also the minimum packet size of 1 or 2 flits.

In current products, the ordering requirement placed upon HOM messages is as follows. For any given caching agent (source) and home agent (destination) combination, messages are kept in order on a per-address basis. Messages from different sources or being sent to different destinations have no ordering requirements respective to each other. Likewise messages to different addresses have no ordering requirements relative to each other. Current home agent designs rely upon the link layer maintaining this HOM channel ordering to ensure proper operation of the coherent protocol.

*Non-Coherent Standard—NCS*

NCS messages are requests dealing with access to system resources that are generally outside of the coherent memory space. This includes things such as reads to non-coherent memory regions, read and write configuration and I/O space transactions, and interrupt acknowledgements. Certain NCS transactions can also access the coherent memory space, but it is not required that coherency be maintained by the Intel QPI hardware in this case. Such read requests would be performed without the addition of snoop requests and responses. Small packets are used, ranging from the minimum packet size, up to three flit packet sizes for some message types. NCS packets do not contain any data flits. However, some messages in NCS support up to an 8 byte payload embedded into the packet's header flits.

71. Intel has had knowledge of the '983 Patent and its infringement of the '983 Patent at least since the filing of the complaint in *VLSI Technology LLC v. Intel Corp.*, Civil Action No. 19-00426 (D. Del.) (filed Mar. 1, 2019) which asserted infringement by Intel of the '983 Patent, and if it did not have actual knowledge prior to that time, it was willfully blind to the existence of the '983 Patent and its infringement of the '983 Patent based on, for example, its publicly-known corporate policy forbidding its employees from reading patents held by outside companies or individuals, as already described above. As still another example, on information and belief, Intel has been sued for infringing patents previously assigned to NXP while this policy was in place. Under the circumstances present here, including explicit notice having been provided of Intel's infringement of other NXP patents and NXP's competitive position with Intel in the marketplace, Intel knew or should have known of the high probability that NXP had patented other technologies, such as those to which the '983 Patent is directed, that Intel had included within its microprocessor products. As yet another example, on information and belief, Kevin Locker, an inventor on the '983 Patent, has been and currently works at Intel as an engineer and as such Intel has been aware of the '983 Patent and its infringement of the '983 Patent since at least the date Mr. Locker began

working at Intel. Intel should have known that its conduct was infringing both prior to and following the filing of VLSI's Delaware Complaint.

72. VLSI is informed and believes, and thereon alleges, that Intel actively, knowingly, and intentionally has induced infringement of the '983 Patent by, for example, controlling the design and manufacture of, offering for sale, selling, supplying, and otherwise providing instruction and guidance regarding the above-described products with the knowledge and specific intent to encourage and facilitate infringing uses of such products by its customers both inside and outside the United States. For example, Intel publicly provides documentation, including datasheets available through Intel's publicly accessible ARK service and software developer's manuals, instructing customers on uses of Intel's products that infringe the methods of the '983 Patent. *See, e.g.*, http://ark.intel.com. On information and belief, Intel's customers directly infringe the '983 Patent by, for example, making, using, offering to sell, and selling within the United States, and importing into the United States, without authority or license, products containing the above-described Intel products.

73. VLSI is informed and believes, and thereon alleges, that Intel has contributed to the infringement by its customers of the '983 Patent by, without authority, importing, selling and offering to sell within the United States materials and apparatuses for practicing the claimed invention of the '983 Patent both inside and outside the United States. For example, the above-described products constitute a material part of the inventions of the '983 Patent and are not staple articles or commodities of commerce suitable for substantial noninfringing use. On information and belief, Intel knows that the above-described products constitute a material part of the inventions of the '983 Patent and are not staple articles or commodities of commerce suitable for substantial noninfringing use. On information and belief, Intel's customers directly infringe the

'983 Patent by, for example, making, using, offering to sell, and selling within the United States, and importing into the United States, without authority or license, products containing the above-described Intel products.

74. As a result of Intel's infringement of the '983 Patent, VLSI has been damaged. VLSI is entitled to recover for damages sustained as a result of Intel's wrongful acts in an amount subject to proof at trial.

75. To the extent 35 U.S.C. § 287 is determined to be applicable, on information and belief its requirements have been satisfied with respect to the '983 Patent.

76. In addition, Intel's infringing acts and practices have caused and are causing immediate and irreparable harm to VLSI.

77. VLSI is informed and believes, and thereon alleges, that Intel's infringement of the '983 Patent has been and continues to be willful. As noted above, Intel has had knowledge of the '983 Patent and its infringement of the '983 Patent. Intel has deliberately continued to infringe in a wanton, malicious, and egregious manner, with reckless disregard for VLSI's patent rights. Thus, Intel's infringing actions have been and continue to be consciously wrongful.

78. Based on the information alleged in this claim, as well as the information alleged in the First Claim *supra* and Third Claim *infra*, VLSI is informed and believes, and thereon alleges, that this is an exceptional case, which warrants an award of attorney's fees to VLSI pursuant to 35 U.S.C. § 285.

## THIRD CLAIM

**(Infringement of U.S. Patent No. 7,793,025)**

79. VLSI re-alleges and incorporates herein by reference Paragraphs 1-78 of its Complaint.

80. The ’025 Patent, entitled “Hardware managed context sensitive interrupt priority level control,” was duly and lawfully issued September 7, 2010. A true and correct copy of the ’025 Patent is attached hereto as Exhibit 3.

81. The ’025 Patent names Robert Ehrlich, Brett W. Murdock, and Craig D. Shaw as co-inventors.

82. The ’025 Patent has been in full force and effect since its issuance. VLSI owns by assignment the entire right, title, and interest in and to the ’025 Patent, including the right to seek damages for past, current, and future infringement thereof.

83. The ’025 Patent states that it “relates generally to interrupt controllers.” Ex. 3 at 1:8-9.

84. The ’025 Patent explains, for instance, that by using a “mode control selector to selectively couple different priority level assignments to a priority encoding module, context sensitive switching of the priority levels assigned to each interrupt request can be implemented with reduced latency.” *Id.* at Abstract.

85. VLSI is informed and believes, and thereon alleges, that Intel has infringed and unless enjoined will continue to infringe one or more claims of the ’025 Patent, in violation of 35 U.S.C. § 271, by, among other things, making, using, offering to sell, and selling within the United States, supplying or causing to be supplied in or from the United States, and importing into the United States, without authority or license, Intel products that use infringing interrupt routing technology.

86. For example, the ’025 accused products embody every limitation of at least claim 1 of the ’025 Patent, literally or under the doctrine of equivalents, as set forth below. The further

descriptions below, which are based on publicly available information, are preliminary examples and are non-limiting.

**[“1. A method for implementing interrupts in a data processing system, comprising:”]**

87. Intel Ivy Bridge processors operate using a method for implementing interrupts in a data processing system.

88. For example, Ivy Bridge processors handle interrupts using the “Power Aware Interrupt Routing” system. *See, e.g.*, Intel Software Developer’s Manual [hereinafter “SDM”] at 2869:

CHAPTER 6
INTERRUPT AND EXCEPTION HANDLING

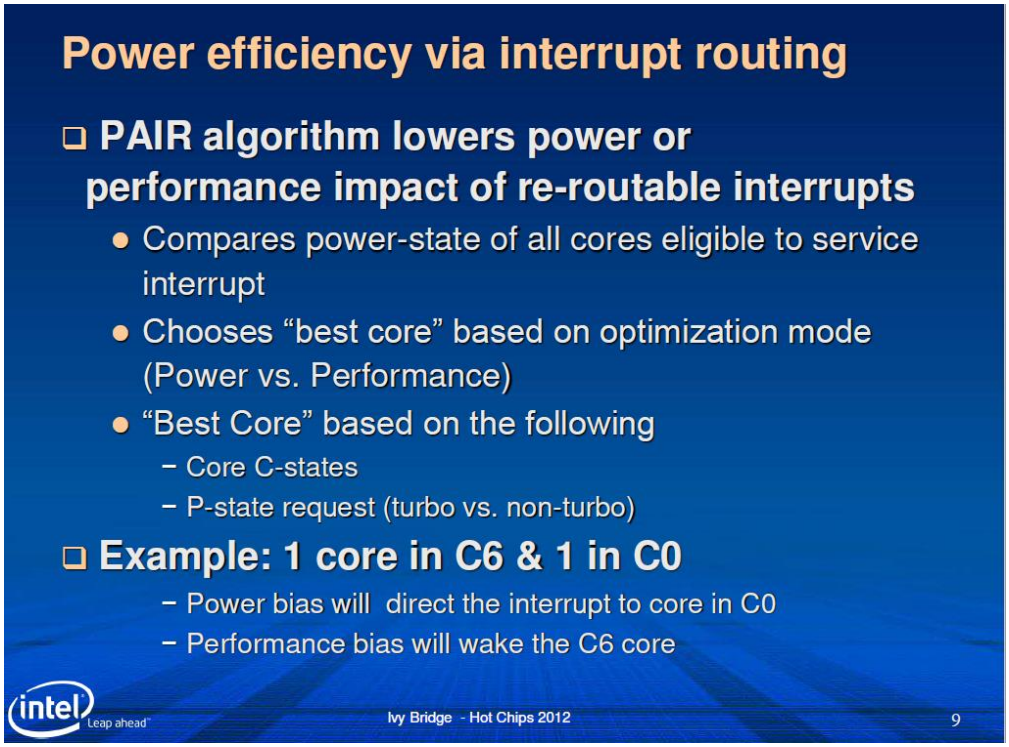
This chapter describes the interrupt and exception-handling mechanism when operating in protected mode on an Intel 64 or IA-32 processor. Most of the information provided here also applies to interrupt and exception mechanisms used in real-address, virtual-8086 mode, and 64-bit mode.

Chapter 20, "8086 Emulation," describes information specific to interrupt and exception mechanisms in real-address and virtual-8086 mode. Section 6.14, "Exception and Interrupt Handling in 64-bit Mode," describes information specific to interrupt and exception mechanisms in IA-32e mode and 64-bit sub-mode.

## 6.1 INTERRUPT AND EXCEPTION OVERVIEW

Interrupts and exceptions are events that indicate that a condition exists somewhere in the system, the processor, or within the currently executing program or task that requires the attention of a processor. They typically result in a forced transfer of execution from the currently running program or task to a special software routine or task called an interrupt handler or an exception handler. The action taken by a processor in response to an interrupt or exception is referred to as servicing or handling the interrupt or exception.

89. *See also, e.g.*, Power Management of the Third Generation Intel Core Micro Architecture formerly codenamed Ivy Bridge [hereinafter "Hot Chips"] at 9:




**["receiving one or more interrupt requests from one or more interrupt sources;"]**

90. Intel Ivy Bridge processors operate using a method that includes receiving one or more interrupt requests from one or more interrupt sources.

91. For example, Ivy Bridge processors can receive one or more interrupt requests by means of, *e.g.*, the INTR pin or through an APIC (Advanced Programmable Interrupt Controller). *See, e.g.*, SDM at 2871:

### 6.3.2 Maskable Hardware Interrupts

Any external interrupt that is delivered to the processor by means of the INTR pin or through the local APIC is called a maskable hardware interrupt. Maskable hardware interrupts that can be delivered through the INTR pin include all IA-32 architecture defined interrupt vectors from 0 through 255; those that can be delivered through the local APIC include interrupt vectors 16 through 255.

**["selectively masking the one or more interrupt requests to generate one or more pending interrupt signals;"]**

92. Intel Ivy Bridge processors operate using a method that includes selectively masking the one or more interrupt requests to generate one or more pending interrupt signals.

93. For example, Intel Ivy Bridge processors can mask the one or more maskable interrupts to generate pending interrupts to the cores. *See, e.g.*, SDM at 2874:

### 6.8 ENABLING AND DISABLING INTERRUPTS

The processor inhibits the generation of some interrupts, depending on the state of the processor and of the IF and RF flags in the EFLAGS register, as described in the following sections.

#### 6.8.1 Masking Maskable Hardware Interrupts

The IF flag can disable the servicing of maskable hardware interrupts received on the processor's INTR pin or through the local APIC (see Section 6.3.2, "Maskable Hardware Interrupts"). When the IF flag is clear, the processor inhibits interrupts delivered to the INTR pin or through the local APIC from generating an internal interrupt request; when the IF flag is set, interrupts delivered to the INTR or through the local APIC pin are processed as normal external interrupts.

94. The selectively masked interrupts are stored as pending interrupt signals to be processed. *See, e.g.*, SDM at 2876:

### 6.9 PRIORITY AMONG SIMULTANEOUS EXCEPTIONS AND INTERRUPTS

If more than one exception or interrupt is pending at an instruction boundary, the processor services them in a predictable order. Table 6-2 shows the priority among classes of exception and interrupt sources.

**["providing a plurality of interrupt priority storage devices comprising a first interrupt priority storage device for storing priority level information associated with a first system mode, and a second interrupt priority storage device for storing priority level information associated with a second system mode; and"]**

95. Intel Ivy Bridge processors operate using a method that includes providing a plurality of interrupt priority storage devices comprising a first interrupt priority storage device for storing priority level information associated with a first system mode, and a second interrupt priority storage device for storing priority level information associated with a second system mode.

96. For example, each of the plurality of cores in Intel Ivy Bridge processors include a "local APIC" (Advanced Programmable Interrupt Control) that stores interrupts. *See, e.g.*, SDM at 3047:

## 10.1 LOCAL AND I/O APIC OVERVIEW

Each local APIC consists of a set of APIC registers (see Table 10-1) and associated hardware that control the delivery of interrupts to the processor core and the generation of IPI messages. The APIC registers are memory mapped and can be read and written to using the MOV instruction.

Local APICs can receive interrupts from the following sources:

- **Locally connected I/O devices** — These interrupts originate as an edge or level asserted by an I/O device that is connected directly to the processor's local interrupt pins (LINT0 and LINT1). The I/O devices may also be connected to an 8259-type interrupt controller that is in turn connected to the processor through one of the local interrupt pins.
- **Externally connected I/O devices** — These interrupts originate as an edge or level asserted by an I/O device that is connected to the interrupt input pins of an I/O APIC. Interrupts are sent as I/O interrupt messages from the I/O APIC to one or more of the processors in the system.
- **Inter-processor interrupts (IPIs)** — An Intel 64 or IA-32 processor can use the IPI mechanism to interrupt another processor or group of processors on the system bus. IPIs are used for software self-interrupts, interrupt forwarding, or preemptive scheduling.
- **APIC timer generated interrupts** — The local APIC timer can be programmed to send a local interrupt to its associated processor when a programmed count is reached (see Section 10.5.4, "APIC Timer").
- **Performance monitoring counter interrupts** — P6 family, Pentium 4, and Intel Xeon processors provide the ability to send an interrupt to its associated processor when a performance-monitoring counter overflows (see Section 18.6.3.5.8, "Generating an Interrupt on Overflow").
- **Thermal Sensor interrupts** — Pentium 4 and Intel Xeon processors provide the ability to send an interrupt to themselves when the internal thermal sensor has been tripped (see Section 14.7.2, "Thermal Monitor").

97. Each local APIC also masks and stores priority information for interrupts. Therefore, each multicore Ivy Bridge processor contains a plurality of interrupt priority storage devices. *See, e.g.*, SDM at 3051:

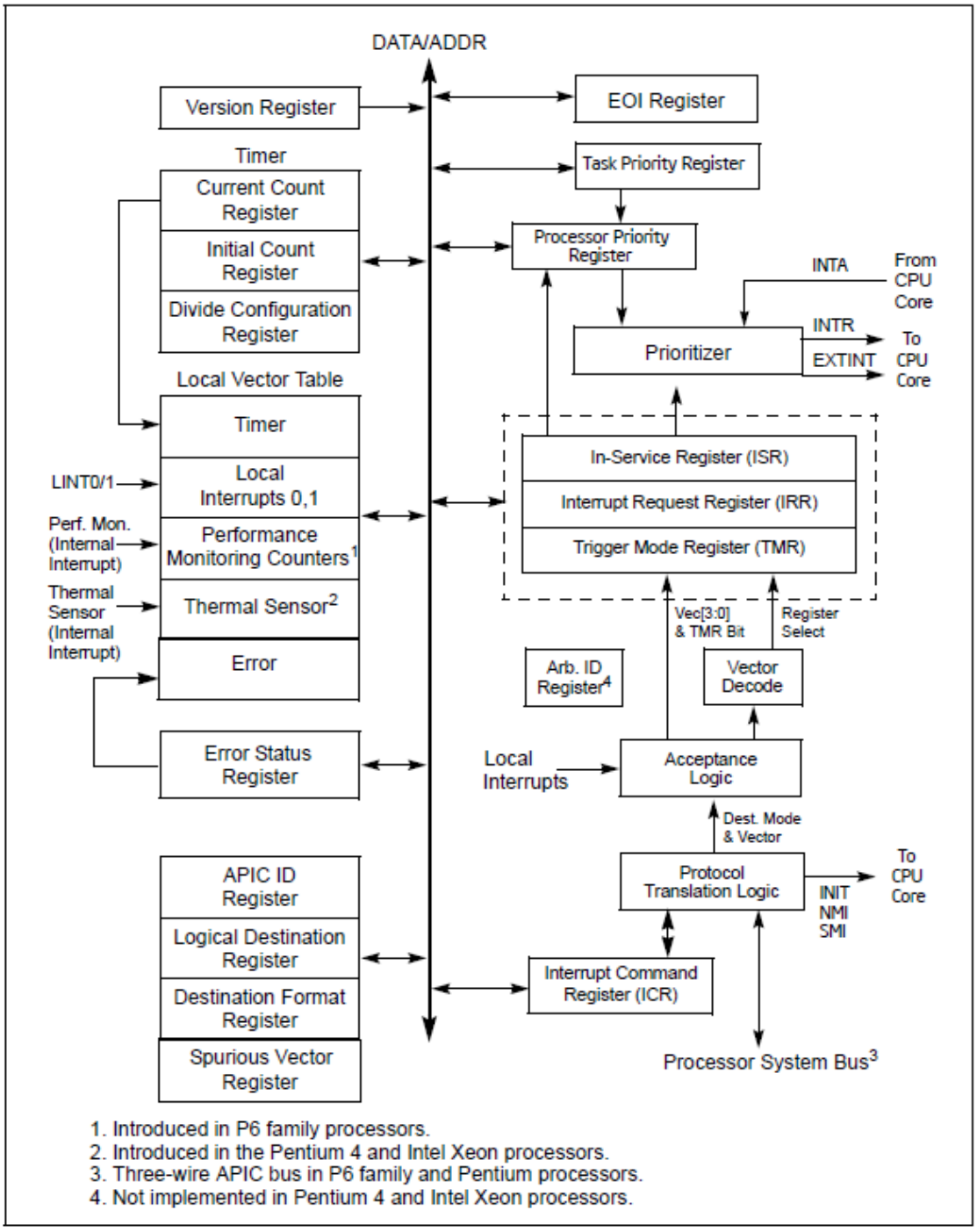
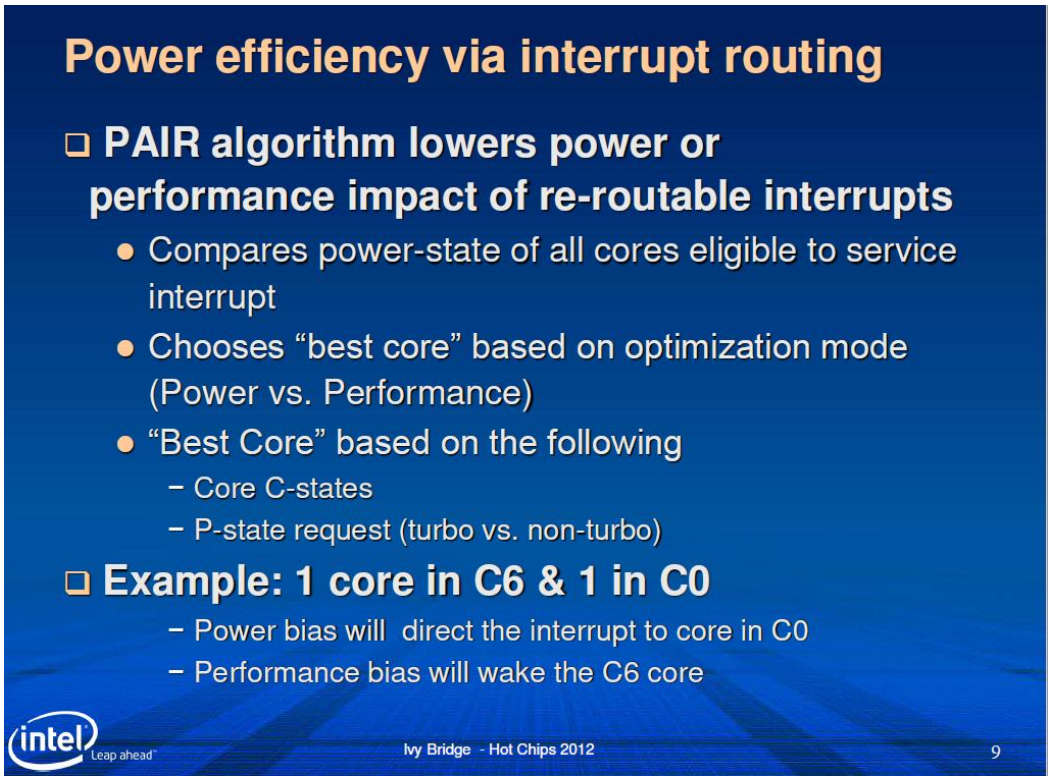



**Figure 10-4. Local APIC Structure**

98. Additionally, interrupts are routed to a particular core depending on parameters such as loading, power state, and power mode. Each core's APIC is associated with a given system mode depending on those parameters. *See, e.g.*, Hot Chips at 9:




**["providing a plurality of interrupt priority storage devices comprising a first interrupt priority storage device for storing priority level information associated with a first system mode for each of the one or more interrupt requests, and a second interrupt priority storage device for storing priority level information associated with a second system mode for each of the one or more interrupt requests; and"]**

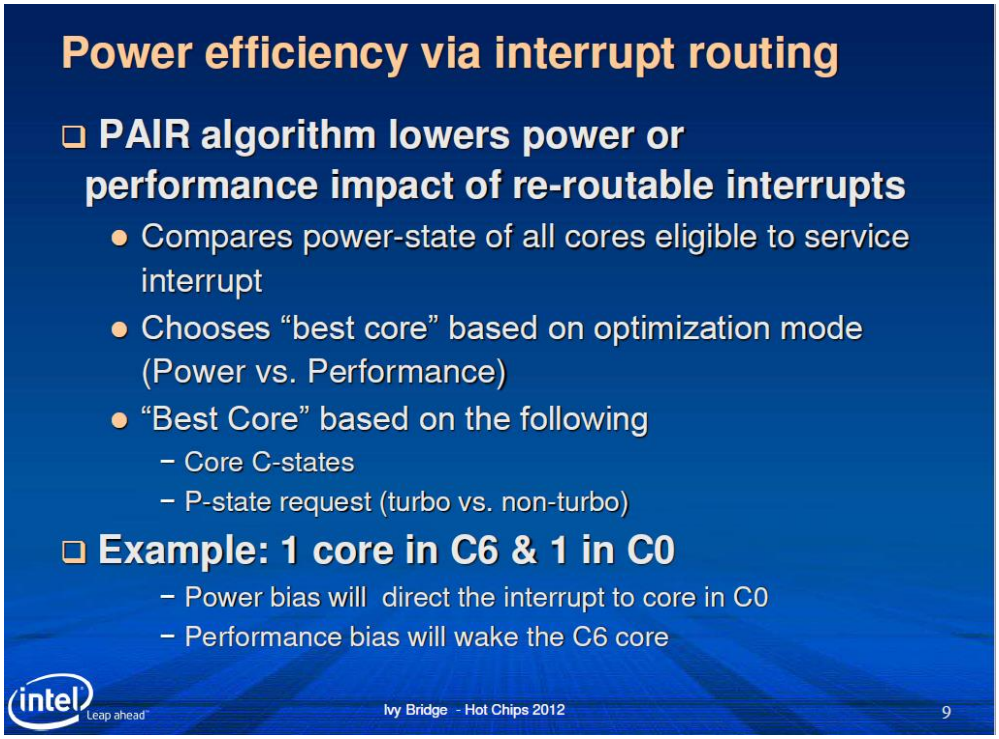
99. Intel Ivy Bridge processors operate using a method that includes providing a plurality of interrupt priority storage devices comprising a first interrupt priority storage device for storing priority level information associated with a first system mode for each of the one or more interrupt requests, and a second interrupt priority storage device for storing priority level information associated with a second system mode for each of the one or more interrupt requests.

100. For example, each local APIC can process events from each of the one or more interrupt requests. See the evidence cited above in connection with the previous element.

**["selectively coupling, in response to a mode control signal, one of the plurality of interrupt priority storage devices to provide logic circuitry with priority level information corresponding to the mode control signal,"]**

101. Intel Ivy Bridge processors operate using a method that includes selectively coupling, in response to a mode control signal, one of the plurality of interrupt priority storage devices to provide logic circuitry with priority level information corresponding to the mode control signal.

102. For example, Intel Ivy Bridge processors selectively couple, in response to power mode and power state control signals, interrupts including priority level information to a given core, and that core's local APIC. *See, e.g.*, Hot Chips at 9. Each core's APIC is associated with a given system mode depending on parameters such as loading, power mode, and power state. *Id.*:



**["where the logic circuitry uses the provided priority level information to prioritize one or more of the pending interrupt signals and"]**

103. Intel Ivy Bridge processors operate using a method where the logic circuitry uses the provided priority level information to prioritize one or more of the pending interrupt signals.

104. For example, the local APIC performs interrupt prioritization. *See, e.g.*, SDM at 3051:

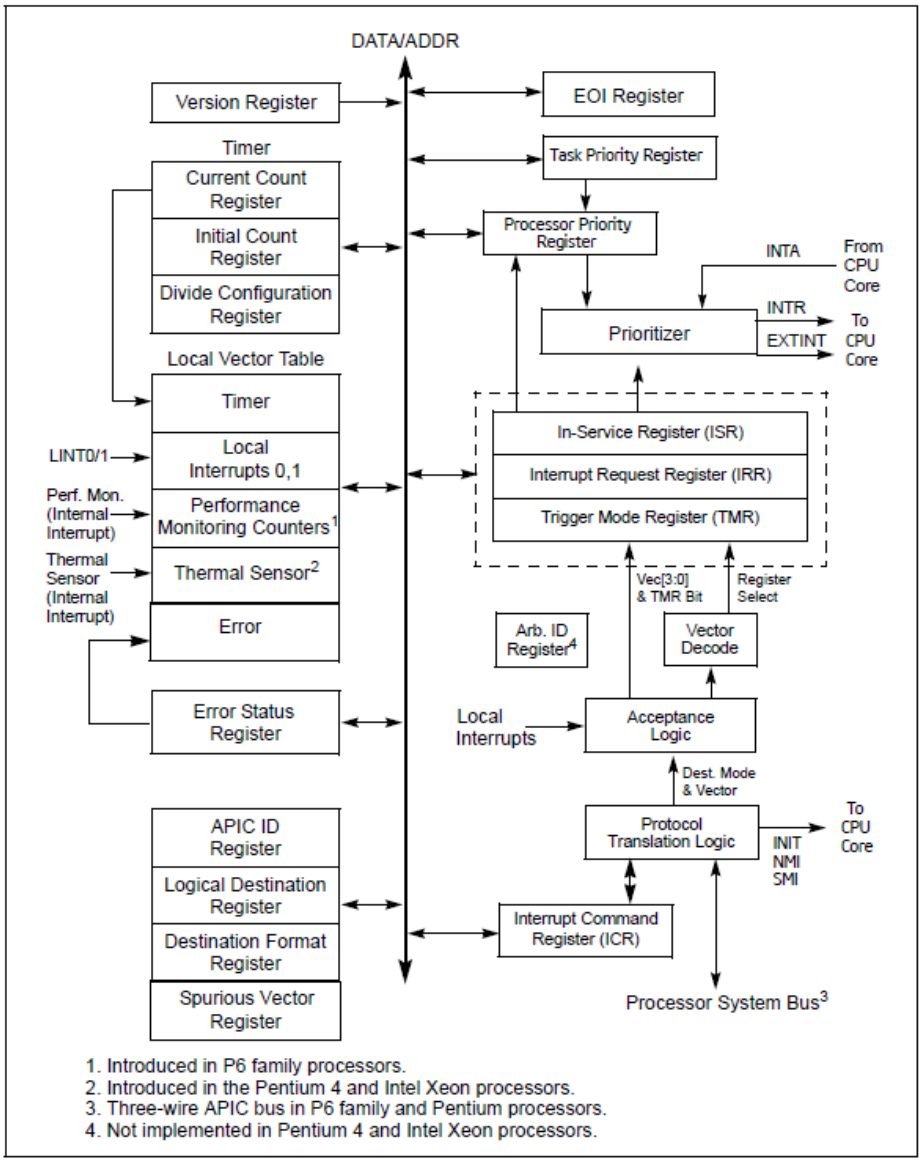
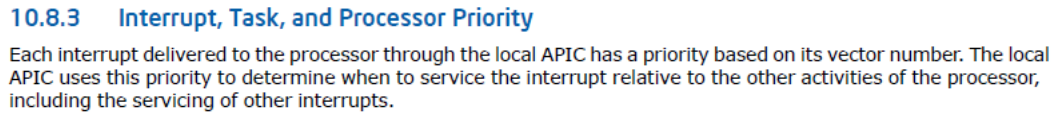



Figure 10-4. Local APIC Structure

105. *See also*, *e.g.*, SDM at 3074:



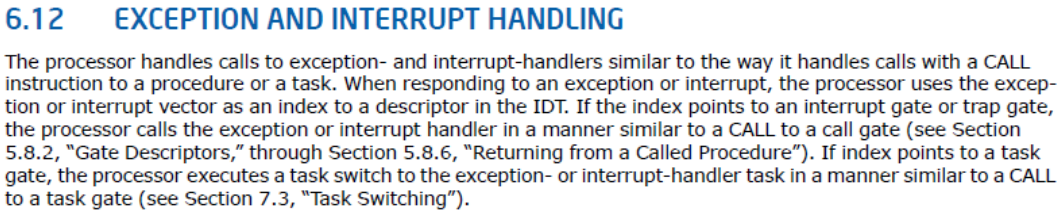
10.8.3 Interrupt, Task, and Processor Priority

Each interrupt delivered to the processor through the local APIC has a priority based on its vector number. The local APIC uses this priority to determine when to service the interrupt relative to the other activities of the processor, including the servicing of other interrupts.

**[“also provides an interrupt request signal which will cause an interrupt to occur in the data processing system.”]**

106. Intel Ivy Bridge processors operate using a method that also provides an interrupt request signal which will cause an interrupt to occur in the data processing system.

107. For example, the local APIC generates an interrupt request signal to a core that causes interrupt handlers to be called in that core. *See*, *e.g.*, SDM at 2879:



6.12 EXCEPTION AND INTERRUPT HANDLING

The processor handles calls to exception- and interrupt-handlers similar to the way it handles calls with a CALL instruction to a procedure or a task. When responding to an exception or interrupt, the processor uses the exception or interrupt vector as an index to a descriptor in the IDT. If the index points to an interrupt gate or trap gate, the processor calls the exception or interrupt handler in a manner similar to a CALL to a call gate (see Section 5.8.2, “Gate Descriptors,” through Section 5.8.6, “Returning from a Called Procedure”). If index points to a task gate, the processor executes a task switch to the exception- or interrupt-handler task in a manner similar to a CALL to a task gate (see Section 7.3, “Task Switching”).

108. *See also, e.g.*, SDM at 3051:

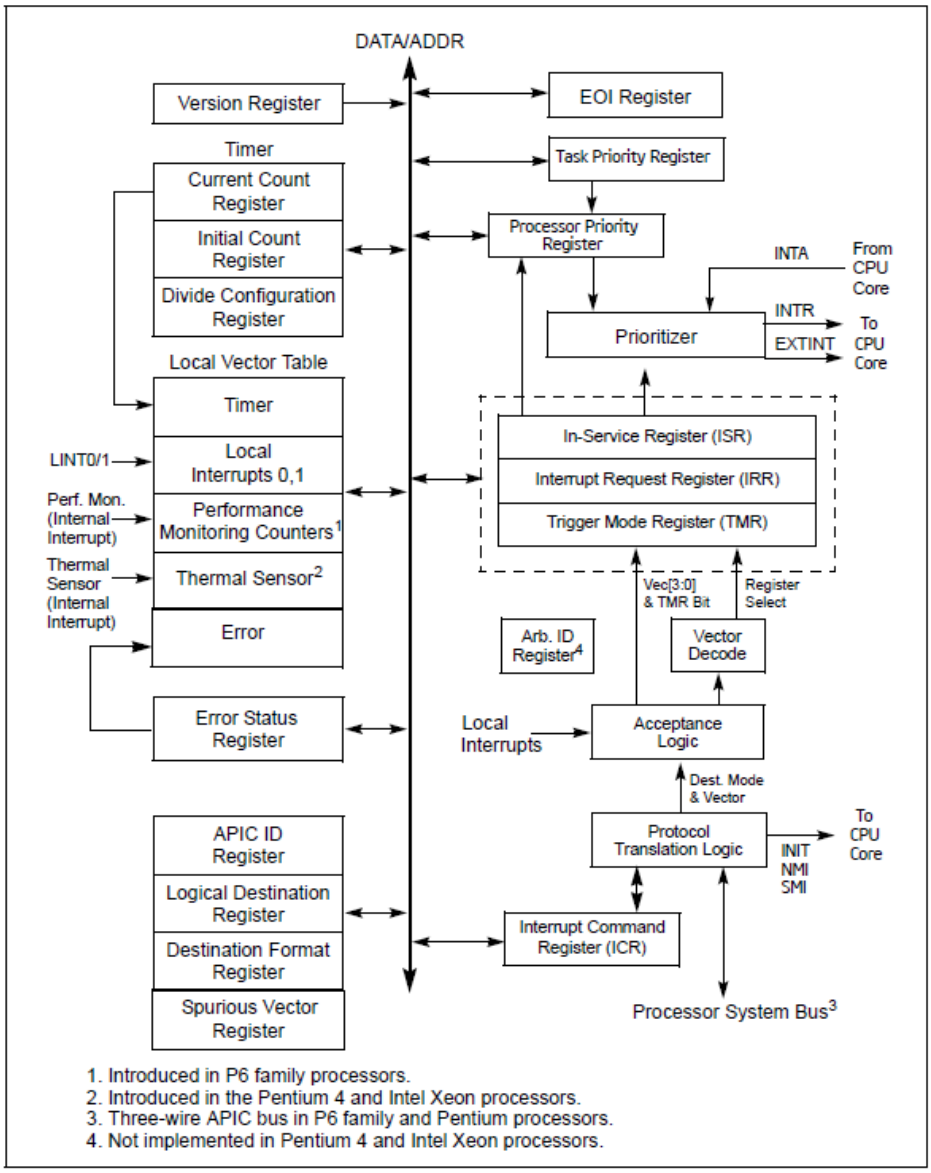



Figure 10-4. Local APIC Structure

109. Intel has had knowledge of the '025 Patent and its infringement of the '025 Patent at least since the filing of the complaint in *VLSI Technology LLC v. Intel Corp.*, Civil Action No. 19-00426 (D. Del.) (filed Mar. 1, 2019) which asserted infringement by Intel of the '025 Patent, and if it did not have actual knowledge prior to that time, it was willfully blind to the existence of the '025 Patent and its infringement of the '025 Patent based on, for example, its publicly-known

corporate policy forbidding its employees from reading patents held by outside companies or individuals, as already described above. As still another example, on information and belief, Intel has been sued for infringing patents previously assigned to NXP while this policy was in place. Under the circumstances present here, including explicit notice having been provided of Intel's infringement of other NXP patents and NXP's competitive position with Intel in the marketplace, Intel knew or should have known of the high probability that NXP had patented other technologies, such as those to which the '025 Patent is directed, that Intel had included within its microprocessor products. As yet another example, on information and belief, Robert Ehrlich, an inventor on the '025 Patent, has been working at Intel since at least November 2016 and currently works at Intel as a Programmable Solutions SoC Architect and as such Intel has been aware of the '025 Patent and its infringement of the '025 Patent since at least the date Mr. Ehrlich began working at Intel. Intel should have known that its conduct was infringing both prior to and following the filing of VLSI's Delaware Complaint.

110. VLSI is informed and believes, and thereon alleges, that Intel actively, knowingly, and intentionally has induced infringement of the '025 Patent by, for example, controlling the design and manufacture of, offering for sale, selling, supplying, and otherwise providing instruction and guidance regarding the above-described products with the knowledge and specific intent to encourage and facilitate infringing uses of such products by its customers both inside and outside the United States. For example, Intel publicly provides documentation, including datasheets available through Intel's publicly accessible ARK service and software developer's manuals, instructing customers on uses of Intel's products that infringe the methods of the '025 Patent. *See, e.g.*, http://ark.intel.com. On information and belief, Intel's customers directly infringe the '025 Patent by, for example, making, using, offering to sell, and selling within the

United States, and importing into the United States, without authority or license, products containing the above-described Intel products.

111. VLSI is informed and believes, and thereon alleges, that Intel has contributed to the infringement by its customers of the ’025 Patent by, without authority, importing, selling and offering to sell within the United States materials and apparatuses for practicing the claimed invention of the ’025 Patent both inside and outside the United States. For example, the above-described products constitute a material part of the inventions of the ’025 Patent and are not staple articles or commodities of commerce suitable for substantial noninfringing use. On information and belief, Intel knows that the above-described products constitute a material part of the inventions of the ’025 Patent and are not staple articles or commodities of commerce suitable for substantial noninfringing use. On information and belief, Intel’s customers directly infringe the ’025 Patent by, for example, making, using, offering to sell, and selling within the United States, and importing into the United States, without authority or license, products containing the above-described Intel products.

112. As a result of Intel’s infringement of the ’025 Patent, VLSI has been damaged. VLSI is entitled to recover for damages sustained as a result of Intel’s wrongful acts in an amount subject to proof at trial.

113. To the extent 35 U.S.C. § 287 is determined to be applicable, on information and belief its requirements have been satisfied with respect to the ’025 Patent.

114. In addition, Intel’s infringing acts and practices have caused and are causing immediate and irreparable harm to VLSI.

115. VLSI is informed and believes, and thereon alleges, that Intel’s infringement of the ’025 Patent has been and continues to be willful. As noted above, Intel has had knowledge of the

’025 Patent and its infringement of the ’025 Patent. Intel has deliberately continued to infringe in a wanton, malicious, and egregious manner, with reckless disregard for VLSI’s patent rights. Thus, Intel’s infringing actions have been and continue to be consciously wrongful.

116. Based on the information alleged in this claim, as well as the information alleged in the First and Second Claims *supra*, VLSI is informed and believes, and thereon alleges, that this is an exceptional case, which warrants an award of attorney’s fees to VLSI pursuant to 35 U.S.C. § 285.

## PRAYER FOR RELIEF

WHEREFORE, VLSI prays for judgment against Intel as follows:

A. That Intel has infringed, and unless enjoined will continue to infringe, each of the Asserted Patents;

B. That Intel has willfully infringed each of the Asserted Patents;

C. That Intel pay VLSI damages adequate to compensate VLSI for Intel’s infringement of each of the Asserted Patents, together with interest and costs under 35 U.S.C. § 284;

D. That Intel be ordered to pay prejudgment and post-judgment interest on the damages assessed;

E. That Intel pay VLSI enhanced damages pursuant to 35 U.S.C. § 284;

F. That Intel be ordered to pay supplemental damages to VLSI, including interest, with an accounting, as needed;

G. That Intel be enjoined from infringing the Asserted Patents, or if its infringement is not enjoined, that Intel be ordered to pay ongoing royalties to VLSI for any post-judgment infringement of the Asserted Patents;

H. That this is an exceptional case under 35 U.S.C. § 285, and that Intel pay VLSI's attorneys' fees and costs in this action; and

I. That VLSI be awarded such other and further relief, including equitable relief, as this Court deems just and proper.

**DEMAND FOR JURY TRIAL**

Pursuant to Federal Rule of Civil Procedure 38(b), VLSI hereby demands a trial by jury on all issues triable to a jury.

Dated: April 11, 2019

Respectfully submitted,

By: */s/J. Mark Mann*

J. Mark Mann
State Bar No. 12926150
mark@themannfirm.com
G. Blake Thompson
State Bar No. 24042033
blake@themannfirm.com
MANN | TINDEL | THOMPSON
300 W. Main Street
Henderson, TX 75652
Telephone: 903.657.8540
Fax: 903.657.6003

Andy Tindel (Texas Bar No. 20054500)
atindel@andytindel.com
**MANN | TINDEL | THOMPSON**
112 E. Line Street, Suite 304
Tyler, Texas 75702
Telephone: (903) 596-0900
Facsimile: (903) 596-0909

Craig D. Cherry (Texas Bar No. 24012419)
ccherry@haleyolson.com
**HALEY & OLSON, P.C.**
100 N. Ritchie Road, Suite 200
Waco, Texas 76701
Telephone: (254) 776-3336
Facsimile: (254) 776-6823

*Attorneys for Plaintiff VLSI Technology LLC*